Oracle® Application Server

Overview and Glossary

Release 4.0.8.1

September 1999

Part No. A60115-03

ORACLE®

Oracle Application Server Release 4.0.8.1 Overview and Glossary

Part No. A60115-03

Primary Author: Alka Srivastava

Contributing Author: Kai Li

Contributors: Steve Button, Francisco Abedrabbo, Sharon Malek, Sanjay Singh, Jim Trezzo, Tom Van Raalte, Sheryl Maring

Contents

Preface v

1 Two and Three-Tier Computing Models

Two-Tier Computing Model 1-1
Three-Tier Computing Model 1-2

2 Oracle Application Server Architecture

HTTP Listener Layer 2-2
Oracle Application Server Layer 2-3
Applications Layer 2-5
Applications and Cartridges 2-5
Applications and Cartridge Servers 2-6

3 Using Oracle Application Server

Creating Applications 3-1
Web Cartridge Applications 3-1
CORBA Applications 3-3
Common Gateway Interface (CGI) Applications 3-6
Deploying Applications 3-6
Configuring Applications 3-7
Application-Level Parameters 3-7
Cartridge-Level Parameters 3-8
Tracing Requests for Applications 3-8
HTTP Requests 3-8

CORBA / IIOP Requests 3-9

4 Benefits of Oracle Application Server

Application Deployment Features 4-1
Wide Range of Application Deployment Solutions 4-2
Development Tools 4-2
Integration of Legacy Applications 4-3
Web-Enabled Databases 4-3
Security Features 4-5
Security Schemes Supported 4-5
Secure Sockets Layer (SSL) Protection 4-6
Oracle Wallet Manager 4-6
System Management Features 4-6
Oracle Application Server Manager 4-6
Scalability 4-7
Failure Recovery 4-7
Monitoring and Site Management 4-8
Load Balancing 4-8
Oracle Application Server Services 4-8
Inter-Cartridge Exchange Service 4-9
Session Service 4-9
Transaction Service 4-9
Logging Service 4-10
Content Service 4-10

Glossary

Index

Preface

Audience

This book is an overview of Oracle Application Server Release 4.0 for a general audience.

The Oracle Application Server Documentation Set

This table lists the Oracle Application Server documentation set.

Title of Book	Part No.
Oracle Application Server 4.0.8 Documentation Set	A66971-03
Oracle Application Server Overview and Glossary	A60115-03
Oracle Application Server Installation Guide for Sun SPARC Solaris 2.x	A58755-03
Oracle Application Server Installation Guide for Windows NT	A58756-03
Oracle Application Server Administration Guide	A60172-03
Oracle Application Server Security Guide	A60116-03
Oracle Application Server Performance and Tuning Guide	A60120-03
Oracle Application Server Developer's Guide: PL/SQL and ODBC Applications	A66958-02
Oracle Application Server Developer's Guide: JServlet Applications	A73043-01
Oracle Application Server Developer's Guide: LiveHTML and Perl Applications	A66960-02
Oracle Application Server Developer's Guide: EJB, ECO/Java and CORBA Applications	A69966-01
Oracle Application Server Developer's Guide: C++ CORBA Applications	A70039-01
Oracle Application Server PL/SQL Web Toolkit Reference	A60123-03
Oracle Application Server PL/SQL Web Toolkit Quick Reference	A60119-03

Title of Book	Part No.
Oracle Application Server JServlet Toolkit Reference	A73045-01
Oracle Application Server JServlet Toolkit Quick Reference	A73044-01
Oracle Application Server Cartridge Management Framework	A58703-03
Oracle Application Server 4.0.8.1 Release Notes	A66106-04

Conventions

This table lists the typographical conventions used in this manual.

Convention	Example	Explanation
bold	**oas.h** **owsctl** **wrbcfg** **www.oracle.com**	Identifies file names, utilities, processes, and URLs
italics	*file1*	Identifies a variable in text; replace this place holder with a specific value or string.
angle brackets	`<filename>`	Identifies a variable in code; replace this place holder with a specific value or string.
courier	`owsctl start wrb`	Text to be entered exactly as it appears. Also used for functions.
square brackets	`[-c string]`	Identifies an optional item.
	`[on|off]`	Identifies a choice of optional items, each separated by a vertical bar (|), any one option can be specified.
braces	`{yes|no}`	Identifies a choice of mandatory items, each separated by a vertical bar (|).
ellipses	n,...	Indicates that the preceding item can be repeated any number of times.

The term “Oracle Server” refers to the database server product from Oracle Corporation.

The term **“oracle**” refers to an executable or account by that name.

The term “*oracle*” refers to the owner of the Oracle software.

Technical Support Information

Oracle Global Support can be reached at the following numbers:

- In the USA: **Telephone: 1.650.506.1500**
- In Europe: **Telephone: +44 1344 860160**
- In Asia-Pacific: **Telephone: +61. 3 9246 0400**

Please prepare the following information before you call, using this page as a check-list:

- ❏ your CSI number (if applicable) or full contact details, including any special project information
- ❏ the complete release numbers of the Oracle Application Server and associated products
- ❏ the operating system name and version number
- ❏ details of error codes and numbers and descriptions. Please write these down as they occur. They are critical in helping WWCS to quickly resolve your problem.
- ❏ a full description of the issue, including:
  - **What** - What happened? For example, the command used and its result.
  - **When** -When did it happen? For example, during peak system load, or after a certain command, or after an operating system upgrade.
  - **Where** -Where did it happen? For example, on a particular system or within a certain procedure or table.
  - **Extent** - What is the extent of the problem? For example, production system unavailable, or moderate impact but increasing with time, or minimal impact and stable.
- ❏ Keep copies of any trace files, core dumps, and redo log files recorded at or near the time of the incident. WWCS may need these to further investigate your problem. For a list of trace and log files, see “Configuration and Log Files” in the *Administration Guide.*

For installation-related problems, please have the following additional information available:

- ❏ listings of the contents of $ORACLE_HOME (Unix) or %ORACLE_HOME% (NT) and any staging area, if used.

- ❑ installation logs (**install.log**, **sql.log**, **make.log**, and **os.log**) typically stored in the **$ORACLE_HOME/orainst** (Unix) or **%ORACLE_HOME%\orainst** (NT) directory.

Documentation Sales and Client Relations

In the United States:

- To order hardcopy documentation, call Documentation Sales: **1.800.252.0303.**
- For shipping inquiries, product exchanges, or returns, call Client Relations: **1.650.506.1500.**

In the United Kingdom:

- To order hardcopy documentation, call Oracle Direct Response: **+44 990 332200.**
- For shipping inquiries and upgrade requests, call Customer Relations: **+44 990 622300.**

Reader's Comment Form

Oracle Application Server Overview and Glossary

Part No. A60115-03

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most about this manual?

If you find any errors or have suggestions for improvement, please indicate the topic, chapter, and page number below:

Please send your comments to:

Oracle Application Server Documentation Manager
Oracle Corporation
500 Oracle Parkway
Redwood Shores, CA 94065

If you would like a reply, please provide your name, address, and telephone number below:

Thank you for helping us improve our documentation.

1

Two and Three-Tier Computing Models

Client / server computing architectures are commonly described as having two or more tiers according to how application logic is distributed between client and server. Minimally, a client / server architecture must have a client tier and a server tier. Oracle's internet computing model is based on a three-tier computing model in which Oracle Application Server functions as a middle tier, or application server tier.

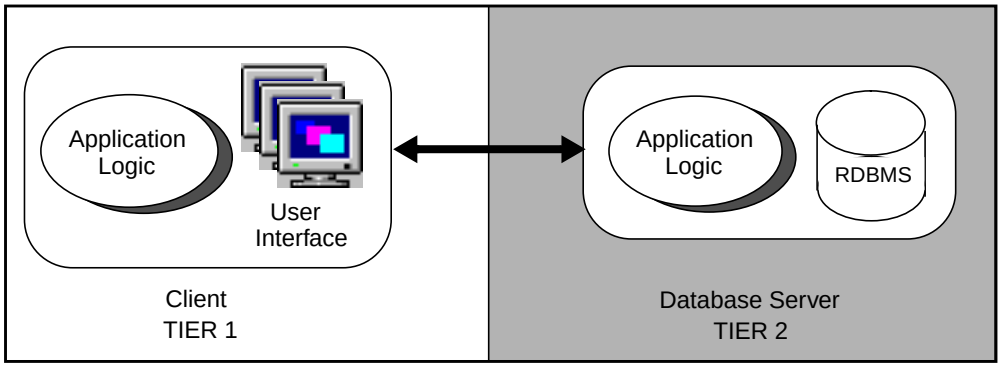
Contents

- Two-Tier Computing Model
- Three-Tier Computing Model

Two-Tier Computing Model

Traditional database client / server architecture is based on a two-tier computing model. This model consists of a client tier and a database server tier (see Figure 1–1). Processing tasks and application logic are shared between the database server and the client.

Figure 1–1 Two-tier computing model



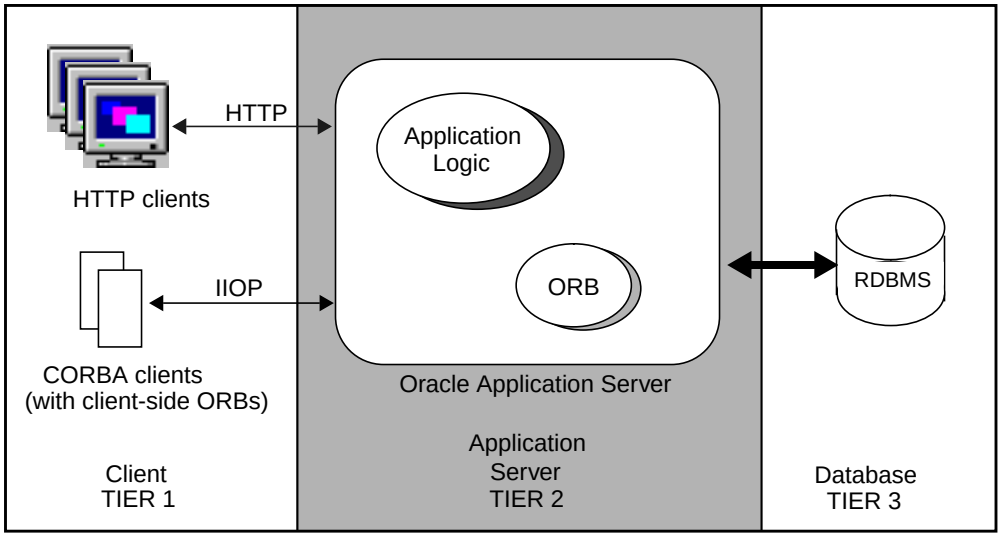
Several disadvantages exist for this model. The clients in a two-tier computing model are fat clients, where much of the processing power and application logic reside. This makes the clients costly to maintain. Furthermore, clients can be operating on different platforms, resulting in the deployment of platform-specific versions of applications.

Three-Tier Computing Model

The three-tier computing model evolved to address the problems of the two-tier model. In a three-tier model, a middle tier exists between clients and the database server. This middle tier consists of an application server that contains the bulk of the application logic. Clients in this model are thin clients. With this architecture, application logic resides in a single tier and can be maintained easily at one point. The architectural design of the middle tier can also be optimized for server functions since it does not have to house the database.

Oracle Application Server serves as the middle tier of the three-tier model as shown in Figure 1–2.

Figure 1–2 Three-tier Oracle Application Server architecture



In this thin-client three-tier architecture, client software (the client tier) is light-weight enough to be downloaded on demand, and does little but present the user interface for a server-side application. The bulk of the application logic is implemented either in the middle (application server) tier or is stored in the database, as in the case of PL/SQL or Java stored procedures.

As the middle tier in a three-tier architecture, Oracle Application Server 4.0 provides the optimal interface between clients and databases. It supports the deployment of distributed component applications based on the Common Object Request Broker Architecture (CORBA) standard. CORBA is an industry-standard model for distributed object-oriented programming and is defined and maintained by an industry consortium called the Object Management Group (OMG) (**http://www.omg.org**).

CORBA specifies the behavior of Object Request Brokers (ORBs). An ORB is responsible for routing requests from clients and other ORBs to CORBA objects. Clients communicate with ORBs using the Internet Inter-ORB Protocol (IIOP) instead of the HTTP protocol. Oracle Application Server 4.0 provides a CORBA 2.0-compatible ORB that clients can use to communicate with server-side applications implemented as CORBA objects.

2

Oracle Application Server Architecture

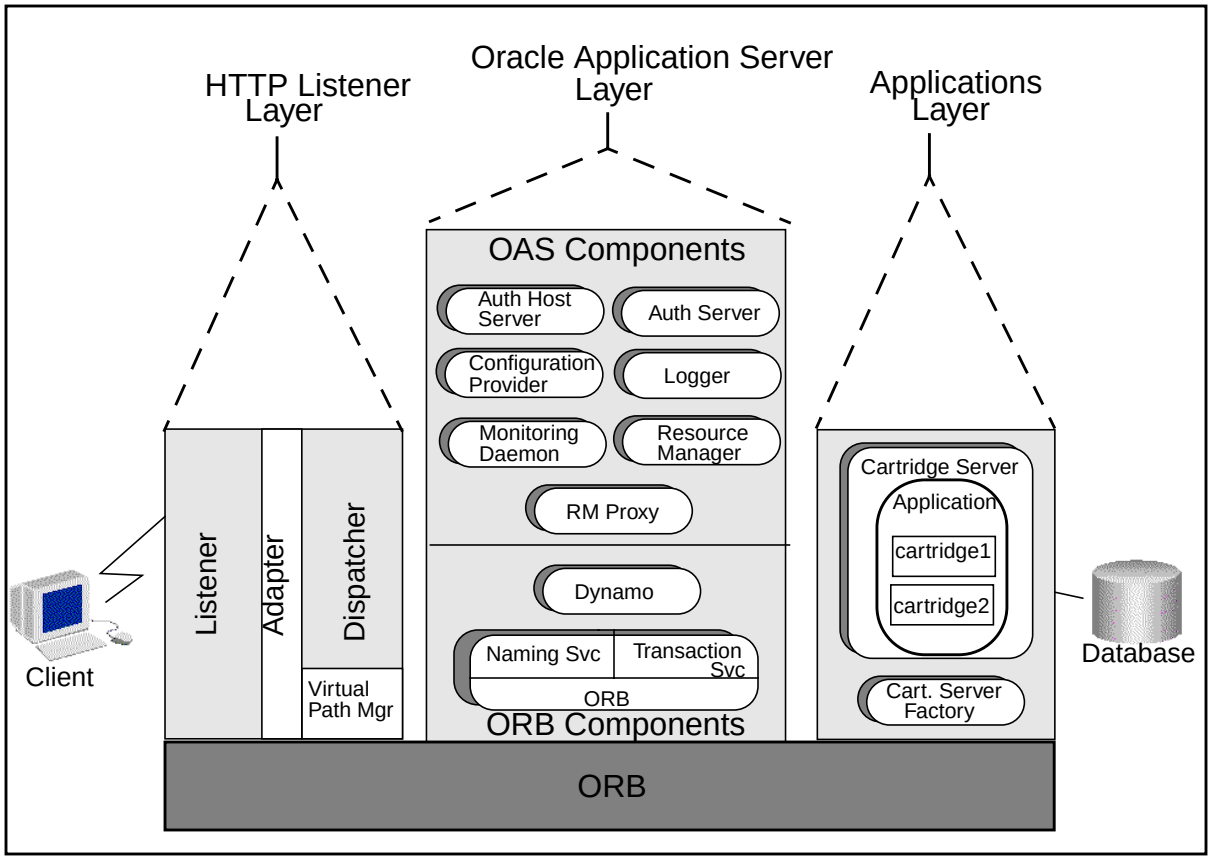
This chapter provides an architectural overview of Oracle Application Server. Oracle Application Server is divided into three layers: the HTTP listener layer, the Oracle Application Server layer, and the applications layer.

Contents

- HTTP Listener Layer
- Oracle Application Server Layer
- Applications Layer

Figure 2–1 shows these layers. Each layer consists of a set of components described below. These layers can optionally be distributed over multiple nodes for scalability and reliability. (See Figure 4–1 on page 4-7).

Figure 2–1 Oracle Application Server architecture



HTTP Listener Layer

The HTTP listener layer is made up of listeners, the adapter interface, and dispatchers.

Listeners

Listeners are HTTP servers; they handle incoming requests and route them to the dispatcher. Oracle Application Server supports the following listeners:

- Oracle Web Listener (included with Oracle Application Server)
- Netscape FastTrack Server

- Netscape Enterprise Server
- Microsoft Internet Information Server (IIS) (Windows NT only)
- Apache (Unix only)

For a list of supported versions, refer to the product release notes.

Oracle Application Server provides adapters for each type of listener to allow listeners to work together with dispatchers in a tightly integrated way, optimizing performance. Note that the documentation refers to the Oracle Application Server listener as the Web Listener and to all other supported listeners as third-party listeners. All of the above listeners are HTTP listeners.

For more information about listeners, see the *Oracle Application Server Administration Guide*.

Adapter

The adapter interface is a common API that both Oracle and third-party listeners use to connect to the dispatcher.

Dispatchers

When a listener receives an HTTP request that does not identify a static HTML page or CGI program, it passes the request to its dispatcher, which assigns a request to a cartridge instance of the appropriate type. This process is described in more detail in "Tracing Requests for Applications" on page 3-8. There is one dispatcher associated with each listener on each node of a Web site.

Virtual Path Manager

The dispatcher forwards requests to the virtual path manager. The virtual path manager maps a request to a cartridge type and passes this information back to the dispatcher. The virtual path manager also passes back authentication requirements to the dispatcher. The dispatcher can then contact the authentication server for authorization and forward the request on to the correct cartridge type.

Oracle Application Server Layer

The Oracle Application Server layer provides resource management in handling requests for applications deployed as cartridges on the server. It provides a common set of components for managing these applications. These components include load balancing, logging, automatic failure recovery, security, directory, and transaction components.

Oracle Application Server components are described in Table 2–1. They run as a single process called **oassrv**. Oracle object request broker (ORB) processes are described in Table 2–2.

Table 2–1 Oracle Application Server components

Component	Description
Authentication Server and Authentication Host Server	Encapsulates the authentication schemes. An Authentication Server consists of one authentication broker and several authentication providers.
Configuration Provider	Reads configuration information from the **wrb.app** and supplies that information to any system component, either remote or local, that needs it. The Configuration Provider resides only on the primary node in a multi-node installation.
Logger	Allows components, including cartridges, to write errors, warnings, or other useful messages to a central repository (a file system or a database).
Monitoring Daemon	Monitors the status of Oracle Application Server processes and objects.
Resource Manager (Broker)	Manages the processes in Oracle Application Server and evaluates and responds to requests. Maintains the correct number of cartridge server processes and cartridge instances. There is one Resource Manager per site.
RM Proxy	Obtains object references to C++, ECO/Java and EJB objects and returns them to clients.

Table 2–2 ORB components

Component	Process	Runs on	Description
ORB	`oasoorb`	Primary node	The main ORB runtime
Naming Service	`oasoorb`	Primary node	The Naming Service is part of the ORB runtime. Returns named objects.
Transaction Service	`oasoorb`	Primary node	The Transaction Service is part of the ORB runtime. Manages operations spanning multiple databases as a single unit.
Dynamo Service	`oasdyn`	Primary node	The process that monitors ORB performance.

Applications Layer

The Applications layer is made up of applications, cartridges, and cartridge servers. For additional information, see "Introduction to Applications" in the *Oracle Application Server Administration Guide.*

Applications and Cartridges

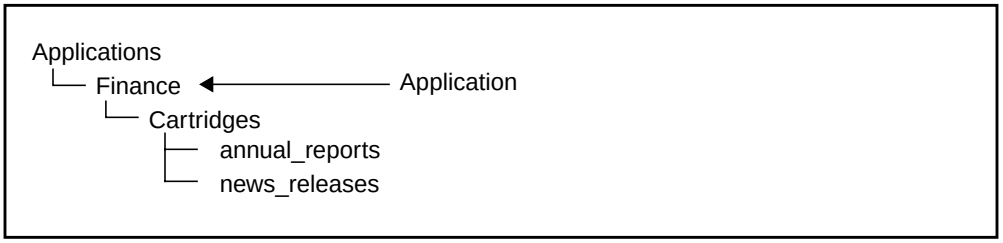
Applications and cartridges are the two main objects that you use when building applications for the application server environment.

A cartridge consists of code that executes application logic and configuration data that enable it to locate the application logic. For example, the PL/SQL cartridge contains code that enables it to connect to Oracle databases and execute PL/SQL stored procedures in the database. The configuration data in the cartridge contains information such as which Oracle database to connect to and what username/password to use in order to run that stored procedure. Cartridges also enable your application to communicate with other components of the application server.

Cartridges can provide runtime environments for specific programming languages. For example, the JServlet cartridge contains a Java Virtual Machine for running Java class files, and the Perl cartridge contains a Perl interpreter for running Perl scripts. See “Creating Applications” on page 3-1 for a list of cartridges.

Cartridges are contained within applications. An application contains one or more cartridges, and within an application, all the cartridges must be derived from the same cartridge type. For example, the following figure shows a PL/SQL application named “finance” that has two PL/SQL cartridges: “annual_reports” and “news_releases”. You cannot have, for example, an application that contains both PL/SQL and JServlet cartridges.

Figure 2–2 A sample application

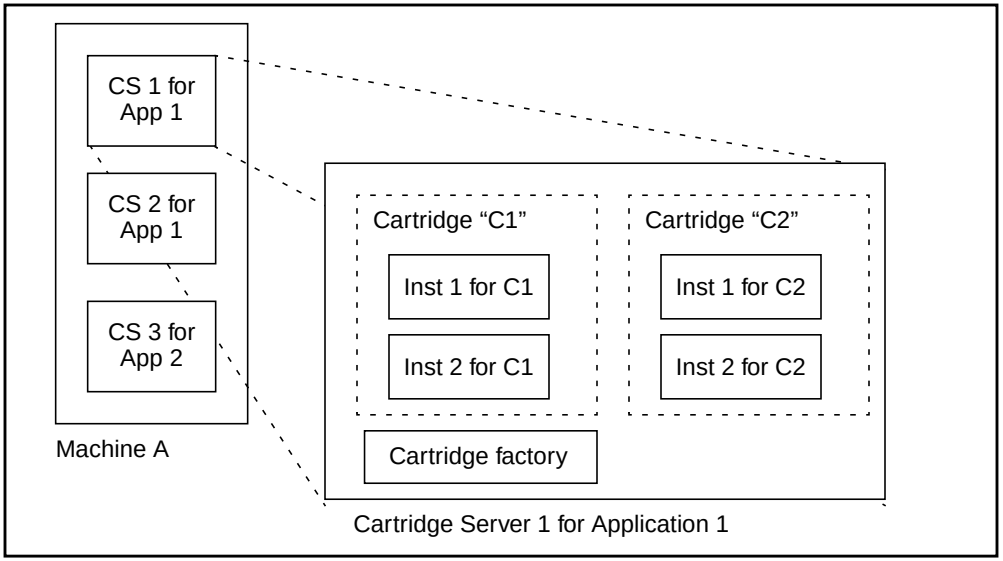


Applications and Cartridge Servers

Each application and its cartridges run within a cartridge server process. Each cartridge server process can run only one application. You can equate applications with cartridge servers.

When an application runs, its cartridge server process instantiates cartridges using the process' cartridge factory object. Once instantiated, the cartridges handle requests from clients. For example, consider the two applications shown in Figure 2–3.

Figure 2–3 Number of cartridge instances and cartridge servers



The figure shows Machine A running three cartridge servers, two for App 1 and one for App 2. In cartridge server 1, which is an instance of App 1, two instances of cartridge 1 and two of cartridge 2 are running. The cartridge factory object in cartridge server 1 instantiates cartridges C1 and C2.

Multiple cartridge servers and multiple cartridge instances are used for load balancing purposes. If you increase the number of clients using your site, you may want to increase the number of cartridge servers and cartridges for your application

3

Using Oracle Application Server

Oracle Application Server provides a platform for running server-side applications. These applications can be written in different programming languages, and can be deployed using different communication protocols. This chapter provides an overview of how you can use Oracle Application Server to run applications. For technical papers and product information go to **http://technet.oracle.com**.

Contents

- Creating Applications
- Deploying Applications
- Configuring Applications
- Tracing Requests for Applications

Creating Applications

Oracle Application Server provides several options for developing applications. Oracle Application Server applications can be divided in the following broad categories:

- Web Cartridge Applications
- CORBA Applications
- Common Gateway Interface (CGI) Applications

Web Cartridge Applications

Oracle Application Server provides a more powerful and efficient kind of server-side application, called a cartridge server. A cartridge server is a shared library that

either implements program logic or provides access to program logic stored elsewhere, such as in a database. A cartridge server is a process in which one or more cartridge instances run. Cartridges and cartridge servers are CORBA objects.

Oracle Application Server provides the following cartridges to develop Web applications. Cartridge-based applications use the HTTP communication protocol.

- PL/SQL Cartridge
- JServlet Cartridge
- LiveHTML Cartridge
- Perl Cartridge
- ODBC Cartridge

PL/SQL Cartridge

The PL/SQL cartridge runs PL/SQL stored procedures in Oracle databases to generate dynamic HTML. The easiest way to embed data from an Oracle database in a Web application is to use the PL/SQL cartridge. The cartridge can also run files that contain PL/SQL source code; it loads the contents of the file into the database and executes the code.

The PL/SQL cartridge comes with the PL/SQL Web Toolkit, which enables you to get information about the request, specify values for HTTP headers such as the content type and cookies, and generate HTML tags.

JServlet Cartridge

The JServlet cartridge contains a Java Virtual Machine and Java class libraries. It provides a runtime environment for server-side Java applications written with the Java Servlet API Specification (available from **java.sun.com**).

JServlets are run on the server side. The cartridge is not involved with running Java applets, which are downloaded and run on the client's machine.

Before invoking a servlet application through the JServlet cartridge, you provide the cartridge with configuration information such as a virtual path, environment variables and authentication information. This configuration is done with the Oracle Application Server Manager.

LiveHTML Cartridge

The LiveHTML cartridge interprets Server-Side Includes (SSI) documents. SSI is a standard that allows you to embed dynamic data in a static HTML document. You

can use special tags in your document to mark the places where the cartridge will substitute dynamic data.

You can also embed Perl scripts in documents for the LiveHTML cartridge to generate dynamic data. This feature lets you generate more complex dynamic data than what is supported by SSI.

The LiveHTML cartridge supports Web Application Objects. This feature enables you to group pages into an "application" object and share data across requests or even between users.

Perl Cartridge

The Perl cartridge runs Perl scripts. This cartridge comes with modules such as database interface API (DBI), DBD::Oracle (for accessing Oracle databases), CGI, and Net.

ODBC Cartridge

The ODBC cartridge allows you to access databases that use the ODBC API. Such databases include Sybase and Informix.

CORBA Applications

Oracle Application Server allows you to develop applications using distributed objects based on the CORBA standard. Communication is based on the IIOP protocol. Oracle Application Server allows you to create CORBA objects that can be accessed from different types of CORBA clients, such as Java applets, Java applications, C++ applications, and CORBA applications.

Oracle Application server supports the following application models:

- Enterprise JavaBeans (EJB) Applications
- Enterprise CORBA Objects (ECO) for Java Applications
- C++ CORBA Cartridge Applications

Enterprise JavaBeans (EJB) Applications

Enterprise JavaBeans (EJB) is a standard introduced by JavaSoft that enables developers to create custom component applications. These applications consist of enterprise beans developed by your company or by third parties. The beans provide the business logic in EJB applications. Oracle Application Server provides support for EJB applications.

EJB applications are flexible to develop: you can use components from different vendors. For example, you can use the configuration and management software from one company, the bean container from another company, and the beans from a third company.

In Oracle Application Server, EJB applications run in a CORBA environment. This means that the beans themselves are CORBA objects and can communicate with other CORBA objects. The containers for the beans are also CORBA objects, and they interact with other components of the application server.

Oracle Application Server allows you to configure EJB applications like you would any other application in the application server. Use the Oracle Application Server Manager to specify values, such as the minimum and maximum number of bean instances, the machines on which the applications can run, and the level of logging that is enabled for the application.

Enterprise CORBA Objects (ECO) for Java Applications

In the Enterprise CORBA Objects (ECO) model, you deploy components written in Java as CORBA components. These components, called ECO objects, are Java classes that you package together to form ECO / Java applications running in an Oracle Application Server environment. An ECO / Java application consists of one or more ECO / Java objects.

Instances of ECOs can be instantiated and accessed from different types of CORBA clients such as Java applets, Java applications, Web cartridges, and CORBA applications.

ECOs provide the business logic for distributed applications, which run on the server and, therefore, do not have a GUI or any other visual representation. The objects define methods that CORBA clients can invoke to perform some operation.

You would generally deploy this kind of application with a client applet, or a Web front end. In this model, the client uses a Web browser to download and run the applet. The applet uses the ORB built into the browser to communicate with the server-side application using IIOP.

ECO/Java vs. EJB Both EJB and ECO / Java are component application models. ECO / Java is an Oracle-proprietary standard that was developed before the EJB specification was finalized by JavaSoft. ECO / Java was created for customers who needed to develop CORBA applications in the Java language. EJB is the Enterprise JavaBeans specification developed by JavaSoft.

Here are some points that differentiate the two models:

Use ECO/Java if:

- You are running traditional CORBA objects (that is, the objects extend `org.omg.CORBA.Object`).
- You do not need to pass objects by value over the network. ECO/Java is based on the CORBA 2.0 specification, which does not support passing objects by value.
- You have clients written in the Java language as well as other languages.

Use EJB if:

- You are running objects that extend the `java.rmi.Remote` interface.
- You need to pass objects by value. (Note that this is not supported in this release of Oracle Application Server.)
- Clients are Java-based. Java is required because the language supports serialized objects, which can be passed by value over the network.
- You want the flexibility of using third-party beans in your application.

For more information about ECO/Java and EJB applications, see the *Oracle Application Server Developer's Guide: EJB, ECO/Java, and CORBA Applications*.

C++ CORBA Cartridge Applications

The C++ CORBA cartridge defines a component architecture for building distributed object-oriented business applications in the C++ programming language. It addresses the development, deployment, and runtime aspects of application development. The C++ CORBA cartridge makes it easy for application developers to write applications. The cartridge runtime shields the developers from low-level details of transactions, multi-threading, resource pooling, etc.

The C++ CORBA cartridges are created and managed at runtime by the C++ CORBA cartridge container, which is provided by Oracle Application Server. The characteristics of the cartridge, such as timeout values, state, etc. are customized at the time of deploying the application. A consistent view of the cartridge is given to the client regardless of how the C++ cartridge is implemented and what functions it provides to the client.

A C++ application consists of one or more C++ cartridges. C++ cartridges typically provide the business logic in C++ applications. The cartridges define methods that clients can invoke to perform some operation.

Common Gateway Interface (CGI) Applications

When an HTTP server receives a request for a CGI application, the server launches the application and uses the CGI to deliver to the application any accompanying query data. Oracle Application Server fully supports this kind of applications.

CGI applications are most useful for processing simple forms on a small scale. Every time the HTTP server receives a request for a CGI application, it starts a new process to run the application. Each CGI application is restricted to running on a single machine. While this approach is adequate for small-scale deployment, it cannot accommodate a large number of clients without seriously degrading performance.

Deploying Applications

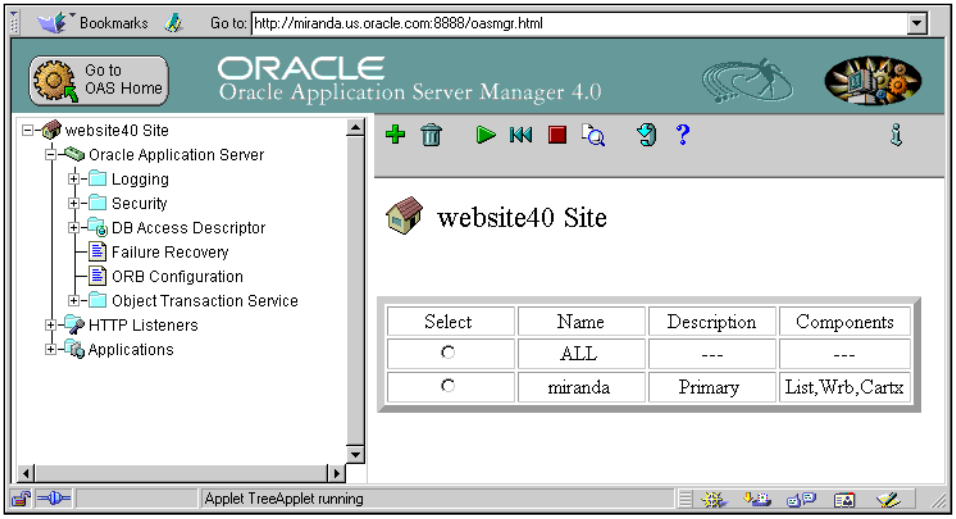
Oracle Application Server Manager is a tool that lets you perform operations to deploy and manage applications, like:

- Add a cartridge to an application.
- Monitor an application by displaying status information for processes associated with that application.
- Administer an application. For example, you can delete, stop, and reload applications.

For more information on using the Oracle Application Server Manager, see the *Oracle Application Server Administration Guide*.

Figure 3–1 displays the Oracle Application Server Manager.

Figure 3–1 Oracle Application Server Manager



Configuring Applications

To configure applications, you will need to configure two types of parameters:

- Application-Level Parameters
- Cartridge-Level Parameters

Application parameters affect the cartridge server processes that run the application, while cartridge parameters affect only individual cartridges.

Application-Level Parameters

Oracle Application Server allows you to set the following application-level parameters:

- Number of seconds a cartridge instance can be idle before it is terminated
- Routing of requests
- Logging parameters
- MIME types that the application handles
- Whether or not clients need certificates in order to access the application
- Whether or not each cartridge instance is bound to a client

- Number of seconds before a session times out
- The HTML page that is returned to the client if an error occurs
- Transaction service for the application

Cartridge-Level Parameters

Cartridge-level parameters vary depending on the cartridge type. For example, if you are developing a PL/SQL application, the cartridge will have different parameters from, say, cartridges in a Perl application.

Oracle Application Server allows you to set the following cartridge-level parameters:

- Virtual path for the cartridge
- Security scheme associated with the virtual path
- Minimum and maximum number of instances for each cartridge
- Minimum and maximum number of threads per cartridge server
- Maximum number of clients that a stateless cartridge can handle

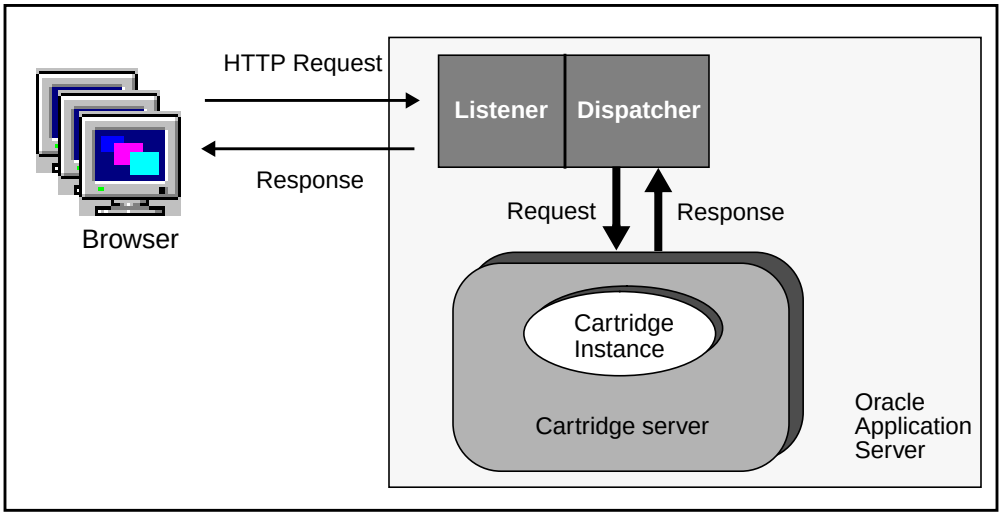
Tracing Requests for Applications

Oracle Application Server can handle both HTTP Requests from Web browser clients, and CORBA/IIOP Requests from CORBA clients by way of a client ORB.

HTTP Requests

Figure 3–2 shows the sequence of events when a Web browser client issues an HTTP request to a server-side application.

Figure 3–2 Life cycle of an HTTP request



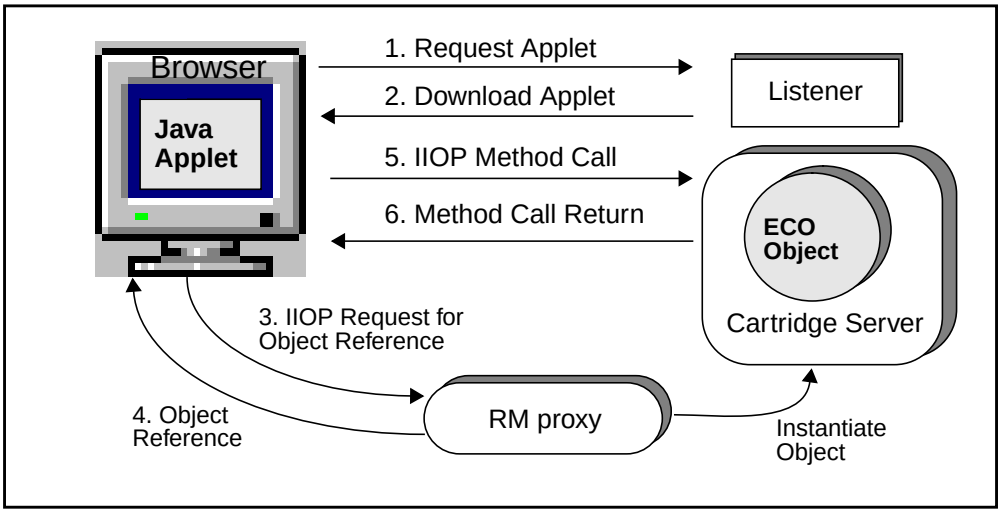
The dispatcher maintains a cache of available cartridge instances. When a request arrives for a particular cartridge, the dispatcher routes the request to one of its cached cartridge instances of the appropriate type.

If the dispatcher has no such cartridge instance available, it requests that Oracle Application Server create a new cartridge in the existing cartridge server. If the existing cartridge server cannot handle any more cartridges, Oracle Application Server creates a new cartridge server and creates a cartridge instance within that cartridge server. The dispatcher then adds this cartridge instance to its cache and assigns the pending request to it. When the request is complete, the new cartridge instance is available to handle a new request.

CORBA/IIOP Requests

Figure 3–3 shows the sequence of events when a Java applet issues a CORBA / IIOP request to a server-side ECO object.

Figure 3–3 Life cycle of a CORBA/IIOP request



When a client requests an object reference, the RM proxy instantiates the requested object (if necessary) within a cartridge server and returns the reference to the client. The client then uses the object reference to call methods on the object directly.

The lifecycle of a request for an EJB application is similar to that of a ECO / Java application.The RM proxy creates an instance of the specified EJB object in an EJB server. For more information on CORBA / IIOP requests, see the *Oracle Application Server Developer's Guide: EJB, ECO/Java and CORBA Applications.*

4

Benefits of Oracle Application Server

This chapter describes the features of Oracle Application Server and how they translate into benefits for the users. For marketing collateral, white papers, and product information go the Website **http://technet.oracle.com.**

Contents

- Application Deployment Features
- Security Features
- System Management Features
- Oracle Application Server Services

Application Deployment Features

Oracle Application Server offers the following features relevant to application deployment:

- Wide Range of Application Deployment Solutions
- Development Tools
- Integration of Legacy Applications
- Web-Enabled Databases

Wide Range of Application Deployment Solutions

Oracle Application Server supports the following types of applications:

- CORBA applications
  - Enterprise JavaBeans (EJB)
  - ECO/Java
  - C++ CORBA
- Web cartridge applications
  - PL/SQL cartridge
  - JServlet cartridge
  - LiveHTML cartridge
  - Perl cartridge
  - ODBC cartridge
- Common Gateway Interface (CGI) applications

See "Creating Applications" on page 3-1 for more on the various options available for developing applications.

Development Tools

Oracle offers a wide range of tools to assist developers in building applications for Oracle Application Server. These tools are:

- Oracle tools
- Third-party tools bundled with Oracle Application Server.

Oracle Tools

Oracle tools for developing applications for Oracle Application Server include:

- Oracle JDeveloper - A Java Integrated Development Environment (IDE) for developing CORBA, EJB, or JServlet cartridge applications.
- Oracle Developer - An IDE for developing applications for the PL/SQL cartridge.

Third-Party Tools

Several third-party tools are bundled with Oracle Application Server. For a complete list, go to the Website **http://technet.oracle.com.**

Integration of Legacy Applications

The Oracle Application Server has a number of features that enable integration of legacy applications.

- pl2java: allows existing PL/SQL stored procedures to be wrapped as Java classes making them available to Oracle Application Server Java applications.
- Transparent and procedural gateways: enable applications to access legacy data, such as data stored in flat files and non-Oracle relational databases.
- CORBA-based technologies: CORBA standards allow software written in various languages to communicate with each other. Thus, a legacy application can be reused simply by defining its interface along CORBA standards.
- COBOL cartridge: allows use of application written in the COBOL programming language. The COBOL cartridge is a third-party product bundled with Oracle Application Server.

Web-Enabled Databases

Oracle Application Server provides mechanisms through which customers can Web-enable their databases. Thus, users can access and update databases through browsers.

Table 4–1 lists the components of Oracle Application Server that you can use to Web-enable different databases.

Table 4–1 Web-enabled database application models

Database	Oracle Application Server Component	Benefits
Oracle	PL/SQL cartridge	Application logic stored in databases as stored procedures. Re-use existing stored procedures with minor changes for HTML display. Scalable and secure performance solution. Supports configurable enterprise transaction services, such as transactions for Web clients. Easy toolset for generating HTML. Web-enable databases leveraging existing skills.
	pl2java	Reuse existing PL/SQL packages in Oracle databases without any modification. Application logic in both PL/SQL and Java. No HTML required. Wizard support in JDeveloper.
Oracle and ODBC-compliant databases	JServlet cartridge (JDBC, SQL, SQLJ)	JDBC access to databases. Application logic in Java. Output HTML using JServlet toolkit. pl2java generated Java wrapper classes enables access to PL/SQL stored procedures. Integrated development and deployment wizards. Integrated with JDeveloper. Support for JTS.
	EJB and ECO/Java (JDBC, SQL, SQLJ)	Industry-standard distributed component models. CORBA development and deployment platform of choice. Enables CORBA developers to access Oracle databases. Database access using JDBC. Access to transaction services such as JTS.

Table 4–1 Web-enabled database application models

Database	Oracle Application Server Component	Benefits
	LiveHTML cartridge (SQL embedded in HTML pages)	Easy to write. Application logic in HTML page.
	Perl cartridge	Easy access through scripts. CGI developers can leverage existing code and add new functionality.
	ODBC cartridge: SQL access to databases	Industry standard. Easy to use.
	COBOL cartridge (bundled third-party tool)	Leverages existing skills. Leverages existing applications.
RDB	RDB stored procedures	Fastest and easiest way to access RDB databases.
Mainframe / Legacy	OpenConnect cartridge and server (bundled third-party tools)	Web-enable data in legacy databases

Security Features

Oracle Application Server provides authentication and encryption capabilities that protect applications and static HTML pages from unauthorized users.

Security Schemes Supported

Oracle Application Server supports the following security schemes:

- Basic, digest, basic_oracle, and crypt authentication schemes. These authentication schemes prompt the user to enter a username and password before the requested page is returned or the application is run.
- IP address and domain name restriction schemes. These restriction schemes allow only authorized machines to access the page or application.
- Certificate authentication scheme. The application server connects to a directory server to verify clients against certificates in the directory server.

  Oracle Application Server Release 4.0 Enterprise Edition supports the use of a Lightweight Directory Access Protocol (LDAP) directory for certificate-based authentication.

For more information on authentication, authorization, and security, see the *Oracle Application Server Security Guide.*

Secure Sockets Layer (SSL) Protection

Secure Sockets Layer (SSL) is a security protocol that supports encryption, integrity, and authentication. Oracle Application Server supports Secure Sockets Layer (SSL) version 3.0.

SSL works at the transport level, which is one level below the application level. This means that SSL can encrypt and decrypt messages before they are handled by application-level protocols such as Telnet, FTP, and HTTP.

SSL encryption and authentication can be used for static HTML pages, CGI scripts, and cartridges. If a request for a cartridge uses SSL, then the resulting HTML page generated by the cartridge is encrypted.

Oracle Wallet Manager

The Oracle Wallet Manager is a Java application that security administrators use to manage public-key security credentials on clients and servers.

Public-key cryptography requires that parties who want to communicate in a secure manner possess certain security credentials. This collection of security credentials is stored in a wallet. A wallet is an abstraction used to store and manage security credentials for the client or the server.

For more information about Oracle Wallet Manger, see the *Oracle Application Server Security Guide.*

System Management Features

Oracle Application Server offers the following features relevant to system management.

Oracle Application Server Manager

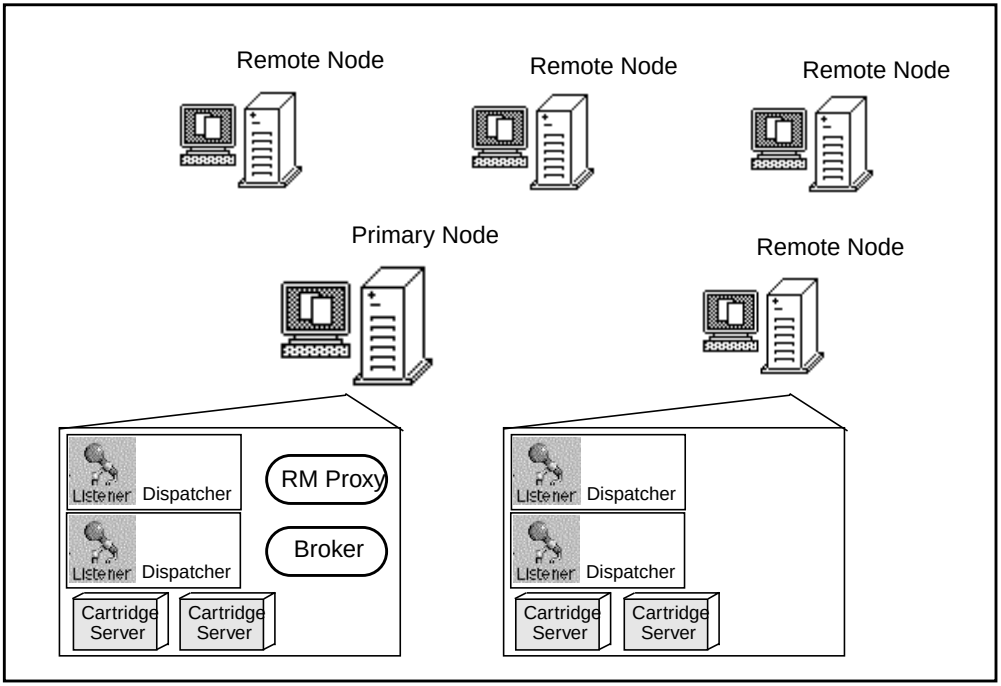
Oracle Application Server uses the Oracle Application Server Manager, a tool for the configuration and management of Oracle Application Server. This interface uses both HTML forms and Java navigational applets to allow an administrator to maintain an Oracle Application Server site. Oracle Application Server also supports integration with Enterprise Manager.

For more information on the Oracle Application Server Manager, see the *Oracle Application Server Administration Guide.*

Scalability

You may set up a single-node or a multi-node installation of Oracle Application Server. When load increases, additional nodes may be easily added to expand the system. In a multi-node installation, ORB processes, the Resource Manager, Configuration Provider, and Monitoring Daemon processes run only on the primary node. All other Oracle Application Server processes, listener processes, and cartridge processes may run on both the primary node and on remote nodes. (See Figure 4–1 and Table 2–1.)

Figure 4–1 Multi-node installation



Failure Recovery

Oracle Application Server provides enhanced mechanisms for detecting and recovering from failures of individual components, such as listeners or cartridge servers.

Rather than putting the burden on a single component to do the failure detection and recovery of all components, Oracle Application Server uses a distributed, self-monitoring failure detection and recovery mechanism. When a component fails,

Oracle Application Server detects the failure and restarts the failed component, restoring any preserved state information when possible.

Monitoring and Site Management

Oracle Application Server provides tools and built-in support for monitoring your site, listeners, and applications. Applications can use the Logging service API to record information in log files. Oracle Application Server also provides support for Common Logfile Format (CLF) and Extended Logfile Format (XLF) system message formats. The Oracle Application Server Manager provides tools for analyzing log files and for tracking and viewing statistics for specific sites, listeners, and applications. These tools also allow you to generate reports on these statistics.

For more information about site management, see the *Oracle Application Server Administration Guide*.

Load Balancing

Oracle Application Server allocates system resources and prioritizes requests based on two types of load balancing methods: priority based and min / max.

Priority-based tuning manages and allocates your system resources automatically based on the priority level you set for your applications and cartridges. Set the load balancing priority classification at the application or cartridge level to High, Medium, Low, or Discretionary. The number of processes, threads, and instances is automatically determined based on the request load and priority level of the application and components.

Min / max tuning allows you to manually set configuration parameters for your applications / cartridges. You have to set instances, threads, and clients parameters for each cartridge at the cartridge level.

For more information about configuring load balancing, see the *Oracle Application Server Administration Guide* and the *Oracle Application Server Performance and Tuning Guide*.

Oracle Application Server Services

The Oracle Application Server provides the following services to customers:

- Inter-Cartridge Exchange Service
- Session Service
- Transaction Service

- Logging Service
- Content Service

Inter-Cartridge Exchange Service

The Inter-Cartridge Exchange (ICX) service enables cartridges to issue requests to and receive responses from other cartridges using a transport-independent, stateless protocol that mimics HTTP. You can use ICX in cases where you can isolate common functionality in one cartridge; other cartridges can use ICX to invoke this cartridge.

Session Service

The session service allows cartridges to be set up so that successive requests from a particular client are handled by the same cartridge instance. This allows you to write your cartridge to maintain state across requests submitted by a specific client.

For example, if you are creating a shopping cart cartridge, you can enable sessions in the cartridge so that it can remember the items that the client has in his or her shopping cart. Each client has its own cartridge instance, and cartridge instance data is not shared across clients.

You can enable and manage sessions in two ways:

- Using the Oracle Application Server Manager. This is called “configurable sessions.”
- Using the session API to manage sessions. This is called “programmatic sessions.” Programmatic sessions give you more control than configurable sessions.

Sessions work with all listeners supported by Oracle Application Server. For more information about the Session service, see the Oracle Application Server Developer’s Guide for the particular development environment you are interested in.

Transaction Service

The transaction service of Oracle Application Server enables applications to perform distributed transactions. The transaction service begins, commits, and rolls back transactions, and keeps track of operations that are within the transaction scope.

You need to use the transaction service along with a database access API, such as OCI or JDBC:

- The database access API parses and executes SQL statements, and gets the execution results.
- The transaction service begins, commits, and rolls back transactions, and establishes connections to the database.

The transaction service implements an extended version of the Object Transaction Service (OTS) model using Java Transaction Service (JTS). OTS is defined by the Object Management Group (OMG), and JTS is defined by JavaSoft. In addition, the application server also supports the Distributed Transaction Processing model, defined by The Open Group.

The transaction model that you use depends on the cartridge you are using. For example, if you are developing applications using the JServlet cartridge, you would use the JTS APIs.

For more information about the transaction service, see the Oracle Application Server Developer's Guides.

Logging Service

The logging service enables each component of Oracle Application Server to log messages and name-value pairs to a log file or database. The logging service enables you to:

- Generate statistics about HTTP requests.
- Trace or debug each component of Oracle Application Server with system messages.
- Generate statistics about each component from the client-defined attributes.

Content Service

The content service of Oracle Application Server provides an API that applications can use to store persistent content data in an Oracle database. Storing content data in a database enhances scalability by allowing multiple cartridge servers running on multiple hosts to access the same content, while relying on the database to resolve access contention.

The PL/SQL cartridge allows you to transfer files from a client machine to a database (uploading) and vice versa (downloading). You can upload and download text and binary files. For more information about the upload/download feature of the PL/SQL cartridge, see chapter 4, "Using the PL/SQL Web Toolkit", of the *Oracle Application Server Developer's Guide: PL/SQL and ODBC Applications.*

Glossary

A B C D E F G H I J K L M N O P R S T U V W X

A

access control list

See ACL.

ACID characteristics

A transaction has ACID characteristics (Atomic, Consistent results, Isolated, and Durable), if it demonstrates the following:

- If interrupted by failure, all effects are undone or rolled back (Atomic).
- The effects of a transaction preserve invariant properties (Consistent results).
- Its intermediate states are not visible to other transactions. Transactions appear to execute serially, even if they are performed concurrently (Isolated).
- The effects of a completed transaction are persistent; they are never lost (except in a catastrophic failure) (Durable).

See the *Oracle Application Server Administration Guide* for further information.

ACL

Access Control List. A list of groups and users authorized for specific access to an object. Or a list of entities, together with their access rights, which are authorized to have access to a specified resource.

adapter layer

The interface between the dispatcher and an HTTP listener. The adapter layer directly interfaces with the native API of an HTTP listener to link the listener to the dispatcher.

advanced networking option

See ANO.

ANO

Advanced Networking Option. Oracle Advanced Networking Option operates within the framework of the Network Computing Architecture to transparently secure the network layer for all applications using SQL*Net and Net8. The option extends client/server connectivity in the distributed environment to embrace network services for enterprise-wide communications.

Apache

A public domain HTTP server derived from the National Center for Supercomputing Applications.

applet

A Java program that runs from an applet viewer or a Web page. In Oracle Application Server, applets can be clients of ECO/Java and Enterprise JavaBeans (EJB) applications.

application

A collection of cartridges that provides business logic. An application contains one or more cartridges, and within an application, all the cartridges must be derived from the same cartridge type since applications provide the runtime environment for cartridges. See the Oracle Application Server Developer's Guides for more information. See also cartridge.

authentication

Authentication ensures that access to static pages, CGI scripts, and cartridges is limited to authorized users. When access to an application or document is protected by an authentication scheme, the client sends identification information to the server, which checks if the client is authorized to access the object. Oracle Application Server supports the following types of authentication:

- basic authentication
- digest authentication

- basic_oracle authentication
- certificate authentication
- crypt authentication

See also security scheme, restriction.

authentication broker

The portion of the authentication server that responds to and evaluates authorization requests.

authentication provider

An Oracle Application Server object that specifies all of the realms used to implement a particular security scheme. An authentication provider is a code module that runs within the authentication server and implements a particular security scheme.

authentication server

An Oracle Application Server object that encapsulates the authentication performed against cartridges. An authentication server consists of one authentication broker object and several authentication provider objects.

B

base directory

The directory to which URL-encoded pathnames addressed to a specific port are appended. For example, if the base directory is **/public_html**, the URL **http://www.blob.com/file** is converted to **/public_html/file**.

basic authentication

A username / password-based authentication scheme that does not encrypt passwords when sending them over the Internet. See also digest authentication and authentication.

basic_oracle authentication

An Oracle-specific authentication scheme. In the basic_oracle (or database) authentication scheme, usernames and passwords are stored in an Oracle database, via a DAD, instead of an operating system file as in basic and digest authentication schemes. The authentication server checks to see if the specified user is a valid database user.

browser client

A client that can access static pages, CGI scripts, and cartridges via a URL over HTTP, HTTPS, or IIOP.

C

C cartridge

An Oracle Application Server cartridge that runs C applications. To use this cartridge, you implement callback functions that are invoked by Oracle Application Server. The C cartridge comes with an application programming interface called the WRB API.

CA

Certifying Authority. A trusted third party that vouches for the identity of an individual, company, or server and signs a certificate. For example, VeriSign, Inc. (**http://www.verisign.com**) is such an authority. See also trustpoint.

cartridge

A cartridge consists of code that executes application logic, and also configuration meta data that defines its runtime behavior. An application may need multiple cartridges to handle large amounts of requests.

For example, the PL/SQL cartridge contains code that enables it to connect to Oracle databases and execute PL/SQL stored procedures in the database. The configuration meta data in the cartridge contains information such as which Oracle database to connect to and what username/password to use in order to execute that stored procedure. Cartridges also enable your application to communicate with other components of the application server. See also application.

cartridge factory

An object that starts up cartridge instances within a cartridge server.

cartridge server

A process in which cartridge instances run. Each application and its cartridges run within cartridge server processes. A cartridge server instance can initialize more than one application instance. See application.

cartridge server factory

A process that starts up cartridge servers.

certificate

A specially formatted data item signed by a trusted third party to attest to the validity of the item's information. Public-key certificates use a CA's signature to attest that the enclosed public key belongs to the principal identified by the enclosed name.

certificate authentication

A method of authentication in which clients identify themselves using X.509 v3 certificates. Oracle Application Server reads the certificate and compares the client's certificate with the certificates of the allowed users.

certificate authority

See CA.

certificate revocation list

See CRL.

CGI

Common Gateway Interface. The industry-standard technique for transferring information between a Web server and any program designed to accept and return data that conforms to the CGI specifications.

CLF

Common Logfile Format. An industry-standard format for Web transaction log files. The Web listener uses this format to log transactions. See also info file.

client

A user, software application (such as a browser), or computer that requests the services, data, or processing of another application or computer (the server).

common gateway interface

See CGI.

common logfile format

See CLF.

common object request broker architecture

See CORBA.

Configuration Provider

The Configuration Provider is a Web Request Broker service. It reads configuration information from the **wrb.app** file and supplies that information to any system component, either remote or local, that needs it. The Configuration Provider resides only on the primary node in a multi-node installation.

connect string

The set of parameters, including a protocol, that is used to connect to a specific database instance on the network. Other names for a connect string include: SQL*Net V2 service name and Net8 connect string. See the *Installation Guide* or your SQL*Net V2 or Net8 documentation for more information about connect strings.

content service

A framework for a document repository where cartridges can store, retrieve, and share documents.

cookie

A text string that is stored onto the client browser by the server to maintain state between HTTP calls. Cookies enable applications to store and retrieve information about a client, such as the domain, path, lifetime, and other variables. Cookies can either expire when the user exits the browser or at a date specified by the creator of the cookie.

CORBA

Common Object Request Broker Architecture. An industry standard for allowing code modules called “objects” to communicate with one another regardless of the programming language in which they are written or the operating system on which they are running. With CORBA, objects are managed by the Object Request Broker (ORB). See also ORB.

CORBA cartridge

A cartridge that provides configuration meta data about a CORBA object. The CORBA cartridge exposes an IDL interface to developers. Such a cartridge is accessed via an object reference over IIOP.

CORBA cartridge server

A cartridge server that hosts one or more instances of CORBA cartridges. The implementation of all CORBA cartridge instances in a CORBA cartridge server must be in the same language.

CORBA object

This is a generic term for a server-side program that conforms to the OMG's CORBA specification. Objects can be written in any language, deployed on any machine, and can exist locally or over a wide-area network.

CRL

Certificate Revocation List. A list of certificates that have been revoked before their scheduled expiration date. CRLs only list revoked certificates. When a revoked certificate is past its original expiration date, it is removed from the CRL.

crypt authentication

A method of providing Unix-style encryption of usernames and passwords, and to authorize access to static pages and Web cartridges. Crypt reads usernames and encrypted passwords from a user defined file name, for example /tmp/foo, as opposed to the username and password entries from the **wrb.app** file. Crypt authentication is only available on the Unix platform.

D

DAD

Database Access Descriptor. A set of values that specify how a PL/SQL cartridge connects to an Oracle database to fulfill an HTTP request. Each PL/SQL cartridge is associated with a DAD. The information in the DAD includes the username (which also specifies the schema and the privileges), password, connect-string, error log file, standard error message, and NLS parameters such as NLS language, NLS date format, NLS date language, and NLS currency.

database access descriptor

See DAD.

database authentication scheme

See basic_oracle authentication.

digest authentication

An authentication scheme that does not send passwords over the Internet. Digest authentication is safer than basic authentication, but is not supported by most browsers. See also authentication and basic authentication.

digital signature

A digital code attached to an electronic document that reliably identifies the author or sender, and verifies that the document has not been tampered with.

directory server

Defines a hierarchical view of an organization's employees, units, and other resources.You can protect applications using directory servers by limiting access to the virtual paths of the applications to particular branches in the directory server.

dispatcher

A dispatcher manages a pool of cartridge instances running on one or more cartridge servers. One dispatcher associated with every listener on each node of a Web site. When a listener receives an HTTP request that does not identify a static HTML page or CGI program, it passes the request to its dispatcher, which assigns and dispatches the request to a cartridge instance of the appropriate type.

distributed transaction processing

See DTP.

domain-based restriction

A restriction scheme that allows or denies access to files based on the client machine's domain name. See also restriction or IP-based restriction.

DTP

Distributed Transaction Processing. The protocol that guarantees a two-phase commit when multiple databases are involved in a transaction.

E

EJB

Enterprise JavaBeans. The component-based application model for Java defined by JavaSoft and various vendors including Oracle. It provides most of the system level services such as multi-threading to ease application programming. EJB relies on various standardized enterprise services, such as JNDI, JTS, JDBC, etc. to facilitate application programming and enable EJB objects to be interoperable across various EJB servers. It fulfills the write once, run anywhere paradigm.

EJB application

A framework of deploying CORBA objects written in Java, which adhere to the EJB specification, on Oracle Application Server. Objects deployed in this way have the same management features as cartridges.

EJB container

The component coordinator in an EJB application and one of the key EJB runtime components. In Oracle Application Server, an EJB container "contains" EJB objects by providing services such as transactions, security, concurrency, and locating beans using JNDI. See also JNDI and EJB application.

EJB deployment descriptor

A serialized object that provides information, such as transaction and security policies, about how an EJB application or object should be deployed.

EJB object

The unit of EJB that contains business logic in Java.

EJB server

The process that implements services needed by EJB containers.

encryption

The practice of encoding (encrypting) data in such a way that only an intended recipient can decode (decrypt) and read the data. See also public-key encryption and shared secret-key encryption.

Enterprise JavaBeans

See EJB.

extended logfile format

See XLF.

F

failure recovery

A system of failure detection and recovery in Oracle Application Server. Components of Oracle Application Server monitor each other continuously. When a component fails, Oracle Application Server detects the failure and restarts the failed component, restoring any preserved state information when possible.

G

genreq

An Oracle Application Server utility for generating requests for certificates. You can submit the generated request to a CA to obtain a certificate.

H

HTML

HyperText Markup Language. A markup language used to create Web pages. It provides a set of markup tags for creating hypertext documents for the Web. Oracle Application Server cartridges enable you to generate HTML tags.

HTTP

HyperText Transfer Protocol. The Internet protocol used to manage communication between Web clients (browsers) and servers.

HTTP listener

See listener.

hypertext markup language

See HTML.

hypertext transfer protocol

See HTTP.

I

IBAC

Identity-Based Access Control. The use of digital IDs to control access to a resource.

ICX

Inter-Cartridge eXchange. A service that allows a cartridge to use the services provided by another cartridge. The cartridges can belong to the same or different application.

identity-based access control

See IBAC.

IDL

Interface Definition Language. A standard language for interface specification primarily used for CORBA object interface definition. IDL is declarative and does not reveal the implementation of a CORBA object. CORBA defines standard mappings from IDL to various programming languages.

IIOP

Internet Inter-ORB Protocol. An Internet transport protocol used by CORBA objects to communicate with each other. In the context of Oracle Application Server, IIOP is used by ECO/Java and EJB objects. IIOP is also used between Oracle Application Server components.

info file

A file to which a HTTP Listener logs its transactions on a particular port. There is one info file for each port on which the HTTP Listener accepts connections. The info file is in Common Logfile Format.

inheritance

The process by which one class acquires the methods and properties of another in object-oriented programming methodology.

initial file

The name of the HTML file that Oracle Application Server returns by default when a URL request specifies only a directory name, for example, index.html.

inter-cartridge exchange

See ICX.

interface definition language

See IDL.

Internet inter-ORB protocol

See IIOP.

interoperable object reference

See IOR.

IOR

Interoperable Object Reference. This is a unique string for each CORBA object and is created when an object reference is passed among different ORBs. You can pass this string to a method to determine the actual object reference. See also object reference.

IP-based restriction

A restriction scheme that allows or denies access to files based on the client machine's IP address. See also restriction.

J

JAR file

An archive file format used by the JAR utility and contains the deployment package of an EJB or ECO / Java application. A JAR file contains the deployment descriptors and class files for the beans in an application, the deployment descriptor for the application, and the manifest file.

JAR utility

An archiving utility for creating and extracting JAR files. See JAR file.

JavaBeans

A portable platform-independent component model that enables developers to write reusable components once and run them anywhere. See also Enterprise JavaBeans.

Java IDE

An integrated development environment used for developing and debugging non-Oracle Application Server portions of a Java application or applet.

Java interpreter

A program that interprets and executes Java bytecode independently of a Web browser.

Java Naming and Directory Interface

See JNDI.

Java Transaction Service

See JTS.

Java Virtual Machine

See JVM.

JCO objects

See ECO / Java objects.

ECO/Java deployment information file (ECO.APP)

A text file that contains information about a ECO / Java application. The information includes the name of the application, the objects in the application, and the names of the remote interface classes.

ECO/Java objects

A way of deploying CORBA objects written in Java on Oracle Application Server. Objects deployed this way have the same management features as cartridges.

JDBC

Java DataBase Connectivity. A Java package that provides connectivity to databases from within Java.

JIT compilation

Just-in-time compilation is the process by which the Java Virtual Machine keeps a copy of native code that it generates from bytecode the first time a method is encountered. Subsequently, when the method is run, the JIT uses the native code without having to interpret the method, resulting in a boost in performance.

JNDI

Java Naming and Directory Interface. JNDI consists of a standard set of APIs that provide directory and naming services. Oracle Application Server has a JNDI naming server which clients can use to obtain object references to ECO / Java objects or Enterprise JavaBean objects.

JTS

Java Transaction Service. It provides the services necessary for applications and databases to become part of a transaction. JTS is the Java version of OTS.

just-in-time compilation

See JIT compilation.

JVM

Java Virtual Machine. A virtual machine is an abstract specification for a computing device that can be implemented in different ways, in software or hardware. You compile to the instruction set of a virtual machine much like you would compile to the instruction set of a microprocessor.

The Java Virtual Machine is part of the Java runtime environment responsible for interpreting Java bytecodes. It consists of a bytecode instruction set, a set of registers, a stack, a garbage-collected heap, and an area for storing methods. Java bytecode is executable by any JVM running on any machine.

JServlet cartridge

An Oracle Application Server cartridge in which the application logic is provided in Java and is executed by a Java Virtual Machine running on Oracle Application Server.

JServlet Toolkit

A group of Java classes provided with Oracle Application Server to provide access to the services of the application server and to generate dynamic HTML using Java.

K

key pair

A pair of mathematically related keys (a public key and a private key) associated with a user and used in public-key encryption.

L

LDAP

Lightweight Directory Access Protocol. A protocol that allows clients to access information from a directory server. This protocol enables corporate directory entries to be arranged in a hierarchical structure that reflects geographic and organizational boundaries. In the context of Oracle Application Server, you can protect URLs by associating them with certificates. The certificate provider accesses a directory server, possibly over SSL, to check if the user has a certificate that allows execution of the URL.

light weight directory access protocol

See LDAP.

listener

Listeners are HTTP servers; they handle incoming requests, and route them to the dispatcher.

For more information, see “HTTP Listener Layer” on page 2-2.

LiveHTML cartridge

Oracle Application Server’s implementation of embedded scripts in HTML files. Scripts can be written in programming languages such as Perl. LiveHTML scripts increase the functionality of HTML pages tremendously by adding programming logic such as control structures and database connectivity to the pages. Dynamic data is thus returned to the client browser.

Log file

See info file.

Logger

The Logger is a Web Request Broker service. It allows other components, such as cartridges, to write errors, warnings, or other useful messages to a central repository (a file system or a database).

M

manifest file

A text file that describes the contents of a JAR file. For EJB JAR files, the manifest file identifies it as an “ejb-jar” file. The manifest file also indicates which elements in the JAR file are the EJB deployment descriptors for the components to be deployed. See also JAR file.

master agent

The process on an SNMP-managed node that accepts queries from the management framework and communicates with the elements to be managed in order to answer the query.

memory mapping

The practice of mapping an open file directly into the address space of a HTTP Listener process. This speeds file access and allows multiple clients to access the same file simultaneously without making a separate copy for each client.

metric

A performance statistic, such as uptime or queue.

migrate

(Third-party HTTP servers.) To transfer configuration information from a third-party HTTP server to the Oracle HTTP server. For example, if you currently use a third-party HTTP server, such as Netscape, but you want to use the Oracle HTTP server that ships with Oracle Application Server instead, you can *migrate* your Netscape configuration to the Oracle HTTP server.

(New versions of Oracle Application Server.) To move from a previous version of Oracle Application Server to the current version.

MIME type

Multipurpose Internet Mail Extension type. A message format used on the Internet to describe the contents of a message and defined by the MIME standard. MIME is used by HTTP servers to describe the type of content being delivered. Several RFCs define MIME. See **http://www.oac.uci.edu/indiv/ehood/MIME/MIME.html**.

monitoring daemon

The monitoring daemon is a Web Request Broker service. It monitors the status of Oracle Application Server processes and objects.

multi-node installation

An Oracle Application Server installation that includes one primary node and at least one remote node. You may install listeners and cartridges on remote nodes for load balancing purposes. A multi-node Web site is a set of distributed Oracle Application Server resources that are controlled through the same WRB, which can only reside on the primary node. See also primary node and remote node.

multiport

A single listener that can respond to requests directed at more than one address/port combination. A multiport can be configured to respond to requests differently if they are made to a different address/port combination on the same listener.

Multipurpose Internet Mail Extensions

See MIME type.

N

Node

A host computer with a unique domain name.

See primary node, remote node.

Node Manager listener

The Node Manager listener is the listener which must be running in order to use the Oracle Application Server Manager. It is used to administer all components of Oracle Application Server except the Web Request Broker. A Node Manager listener must be present on each node in your Web site, and is started up automatically when you install Oracle Application Server. This listener cannot execute cartridges and is not configurable.

noun

In the scope of performance monitoring tools, a noun is a component of an Oracle Application Server site. Examples of a noun are component, object, cartridge, class, bean, process, piece of code, computer, process, etc.

O

Object Management Group

See OMG.

object reference

A unique identifier that is used to represent an object instance in a distributed system. See also IOR.

Object Request Broker

See ORB.

Object Transaction Service

See OTS.

OCI

Oracle Call Interface. An API or low-level tool to access Oracle databases and execute SQL and PL/SQL statements.

ODBC

Open DataBase Connectivity. A standard or open application programming interface for accessing a database. The goal of ODBC is to make it possible to access any data from any application, regardless of which database management system (DBMS) is handling the data. ODBC manages this by inserting a middle layer, called a database driver, between an application and the DBMS. The purpose of this layer is to translate the application's data queries into commands that the DBMS understands.

ODBC cartridge

An Oracle Application Server cartridge that allows you to access ODBC databases, such as Informix or Sybase.

OFA

Optimal Flexible Architecture. A directory structure which ensures a logical division of product and configuration files. In this structure, you can easily upgrade product files without affecting configuration files. The basic concept of OFA is to separate data, configuration files, and executables.

OMG

Object Management Group. A consortium with a membership of more than 700 companies. The organization's goal is to provide a common framework for developing applications using object-oriented programming techniques. OMG is responsible for the CORBA specification. Their URL is **http://www.omg.org**.

open database connectivity

See ODBC.

optimal flexible architecture

See OFA.

Oracle Application Server Manager

A collection of HTML forms and Java navigational applets to allow an administrator to configure and maintain an Oracle Application Server site using a browser.

Oracle Call Interface

See OCI.

ORACLE_HOME

Environment variable that indicates the root of the Oracle database code tree. This refers to the place where Oracle software is installed.

Oracle Wallet Manager

A Java-based application that security administrators use to manage public-key security credentials on clients and server. Security credentials consist of a public/private key pair, a certificate, and trustpoint. See also public-key cryptography and wallet.

Oracle Web Agent

See OWA.

ORB

Object Request Broker. In CORBA, an ORB acts as a "broker" between a client request for a service from a distributed object or component and completion of that request. Having ORB support in a network means that a client program can request a service without having to understand where the server is in the distributed network or what language the server object is written in.

OTS

Object Transaction Service. It provides the services necessary for applications and databases to become part of a transaction. See also JTS.

OWA

Oracle Web Agent. A term from an earlier release of this product. OWA is now called the PL/SQL cartridge. For the sake of compatibility, the string "owa" is still used in various places in the product.

P

package

A group of PL/SQL and Java functions and procedures.

Perl cartridge

An Oracle Application Server cartridge that runs Perl scripts. This cartridge comes with modules, such as DBI, DBD::Oracle (for accessing Oracle databases), CGI, and Net.

PL/SQL cartridge

An Oracle Application Server cartridge that interfaces to an Oracle database. The cartridge can run stored procedures within the database and return dynamically generated pages that contain data from the database.

PL/SQL Web Toolkit

A bundle of PL/SQL packages, provided with Oracle Application Server, that let you generate HTML using PL/SQL. Applications written for either the PL/SQL cartridge or the JServlet cartridge can use these packages. Also referred to as the PL/SQL Toolkit.

primary node

Where the WRB and configuration files are stored. The host where the WRB and configuration files are stored is called the primary node in a multi-node configuration. A primary node contains the same components as a single-node, but a primary node has one or more remote nodes associated with it. See also remote node and multi-node installation.

private key

A key used by a limited number of communicating parties to decrypt data encrypted with a public key. See also public-key encryption.

processor sets

Introduced in Solaris 2.6, processor sets allow a group of processors to be allocated for the exclusive use of one or more applications. This gives a system administrator control over the creation, management, and binding of processes into processor sets.

proxy server

A proxy server typically sits on a network firewall and allows clients behind the firewall to access Web resources. All requests from clients go to the proxy server rather than directly to the destination server. The proxy server forwards the request to the destination server and passes the received information back to the client. The proxy server channels all Web traffic at a site through a single, secure port; this allows an organization to create a secure firewall by preventing Internet access to internal machines, while allowing Web access.

public key

A key known to all users, used to encrypt data in such a way that only a specific user can decrypt it. See also private key and public-key encryption.

public-key cryptography

A set of well-established techniques and standards that ensure that data passes securely over a network, from the sender to the receiver, without any interference from a third party. Public-key cryptography facilitates encryption and decryption, tamper detection, authentication, and nonrepudiation.

public-key encryption

A form of encryption that uses a key pair (a public key and a private key) to encrypt and decrypt data.

R

RC4, RC5

RSA encryption algorithms.

realm

A group of users and groups assigned by an authentication scheme to regulate access to specific files or directories.

remote interface

For ECO / Java, a remote interface is used to specify the methods in a ECO / Java object that clients can invoke. This interface is written in Java but is converted to IDL files by Oracle Application Server using a Java-to-IDL reverse mapper. The IDL files are then converted by an IDL-to-Java compiler to Java stub and skeleton classes for use by clients. One advantage of the remote interface is that ECO / Java object developers need not learn IDL to publish their object's methods. See reverse mapper.

remote method invocation

See RMI.

remote node

In a distributed, multi-node installation, you set up one primary node and one or more remote nodes. A remote node may be of one of the following types:

- Cartridge only node
- Listener only node
- Cartridge and listener node

Remote nodes do not run the Resource Manager.

See also primary node.

Resource Manager

See RM.

request latency

The time required to process a request.

request throughput

The number of requests processed per unit of time.

restriction

A security scheme that restricts access to files provided by Oracle Application Server to client machines within certain groups of IP addresses or DNS domains.

See also domain-based restriction and IP-based restriction.

reverse mapper

A utility that converts a CORBA server object's interfaces to IDL (for example, Java to IDL). These IDL interfaces can then be converted to CORBA stub and skeleton code by a IDL compiler (e.g. IDL-to-Java) for use by clients.

RM

Resource Manager. This term is used in two ways.

(Web Request Broker) The RM is a Web Request Broker service that manages the processes in Oracle Application Server, and maintains the correct number of cartridge server processes and cartridge instances. There is one Resource Manager per site.

(Database) The RM is an Oracle7 or Oracle8 database that manages data integrity for transaction resource changes. Multiple RMs can exist in an environment to serve requests.

RMI

Remote Method Invocation. An interaction scheme for distributed objects written in Java. It enables a Java program running on one computer to access the methods of another Java program running on another computer.

RM proxy

The Resource Manager proxy is a Web Request Broker service that obtains object references to ECO/Java and EJB objects and returns them to clients.

S

scalability

The ability to handle increasing numbers of hardware requests without adversely affecting latency and throughput.

scripting

Using a simple programming language to perform tasks. This language usually has limited functionality but can be implemented easily and quickly. In Oracle Application Server, scripts written in a scripting language such as Perl can be embedded in HTML pages to increase the functionality of the pages.

See also LiveHTML cartridge.

secure sockets layer

See SSL.

security scheme

It prevents unauthorized users from accessing protected files or applications. Oracle Application Server supports both authentication schemes and restriction schemes.

Oracle Application Server supports the following types of authentication:

- basic authentication
- digest authentication
- basic_oracle authentication
- certificate authentication
- crypt authentication

The following restriction schemes are supported:

- IP-based restriction
- domain-based restriction

serializable

An interface that a Java class implements to enable itself to be persistent through the Java Serialization service. This service allows a bean's state to be stored and retrieved.

server

There are two types of servers relevant to this product. The first is Oracle Server, which is a relational database server dedicated to performing data management duties on behalf of clients using any number of possible interfaces. The other is Oracle Application Server which is a collection of middleware services and tools that provide a scalable, robust, secure, and extensible platform for distributed, object-oriented applications. Oracle Application Server supports access to applications from both Web clients (browsers) using HTTP and CORBA clients, which use the Common Object Request Broker Architecture (CORBA) and the Internet Inter-ORB Protocol (IIOP).

session key

A secret key used by SSL to encrypt data transmitted over a secure connection. The client generates the session key after Oracle Application Server authenticates itself and sends it to Oracle Application Server using public-key encryption.

shared secret-key encryption

A form of encryption that uses a single key both to encrypt and to decrypt a document. Shared secret-key encryption is much faster than public-key encryption, but is more vulnerable to attack.

Simple Network Management Protocol

See SNMP.

single-node installation

The installation of an application server on a single machine. See multi-node installation.

site

A site is a group of nodes sharing the same Web Request Broker. You must have at least one primary node in your site; you may have remote nodes in your site as well.

site.app

The site.app file contains configuration information about different nodes in the web site. There is one site.app file for each web site. The file is located at: %ORACLE_HOME%\ows\admin*website*\wrb\site.app for NT installations or $ORAWEB_ADMIN/*website*/wrb/site.app for Unix installations.

skeletons

In CORBA, skeletons are classes that are generated as part of the IDL compilation process. They receive calls from client stubs, unmarshal parameters, and call the method implementation. See also IDL and skeletons.

SNMP

Simple Network Management Protocol. A set of protocols for managing complex networks. The versions of SNMP were developed in the early 1980s. SNMP works by sending messages, called protocol data units (PDUs), to different parts of a network. SNMP-compliant devices, called agents, store data about themselves in Management Information Bases (MIBs) and return this data to the SNMP requesters. Oracle Application Server Release 4.0 offers SNMP support.

SSI

Server Side Include. A directive in a LiveHTML page, in the form of a HTML comment, that instructs a web server to provide dynamic data to the page. This data can include information about the operating environment of the server.

See also LiveHTML cartridge.

SSL

Secure Sockets Layer. A standard for the secure transmission of documents over the Internet using secure HTTP (HTTPS). SSL uses digital signatures to ensure that transmitted data is not tampered with.

stored procedure

A set of PL/SQL instructions that are stored in a database.

stubs

In CORBA, stubs are classes that are generated as part of the IDL compilation process (e.g. using an IDL-to-Java compiler). The compiler creates the stubs from the IDL interfaces of a server object. The stubs contain code that handles operations needed to access remote objects. A client application uses the generated stubs to access methods in the remote server object. See also IDL and skeletons.

subagent

A process that receives queries for a particular managed element from the master agent and sends back the appropriate answers to the master agent. The Oracle Application Server SNMP subagent detects unexpected events and forwards them to the master agent.

subscription

In the scope of the performance monitoring tools, a subscription is a request by a user to track a particular metric.

T

third-party HTTP server

See third-party listener.

third-party listener

Oracle Application Server supports HTTP servers such as:

- Netscape FastTrack Server
- Netscape Enterprise Server
- Microsoft Internet Information Server (IIS) (Windows NT only)
- Apache (Unix only)

These HTTP listeners are referred to as third-party listeners.

See also Web Listener, HTTP listener.

threads

A thread in Oracle Application Server is a flow of control in a cartridge instance that is initiated by a client request. A thread allocates program resources in the cartridge instance to the request. A cartridge instance can have more than one thread and multiple threads can share resources.

three-tier architecture

A client-server computing architecture where systems are separated into three layers: the interface layer, the application logic layer, and the database layer. Clients in this model are thin clients which do not contain as many resources as clients in the two-tier architecture. The application logic layer which is also known as the middle-tier provides the computing power and resources for the client. Oracle Application Server fulfills this role ideally as a development and deployment platform for database enabled applications.

TP Monitor

Middleware software used extensively in the mainframe environment for building client/server or distributed applications. Examples of TP Monitors include Tuxedo, CICS, Encina, and Top End.

transaction

A transaction is a collection of operations that work toward a goal of satisfying a single user's need. A transaction guarantees that all operations within it either succeed or fail.

transaction service

An OTS service that enables you to perform transactions that span several HTTP requests. The transaction service is based on the XA open model transactions defined by the X/Open Company.

transactional DAD

A database that has been configured to be involved within a transaction. Once configured, more than one database can be managed by Oracle Application Server in committing resources on all databases involved in the transaction. A database cannot be involved in a transaction unless the DAD is configured to be transactional.

trustpoint

Also referred to as a trusted certificate authority (CA), a trustpoint is an issuer of certificates whom you can trust. You can use Oracle Wallet Manager to manage trustpoint information associated with your application.

two-tier architecture

The traditional client-server computing architecture with application front-end and logic executing on a client that accesses a database.

U

uniform resource identifier

See URI.

uniform resource locator

See URL.

URI

Uniform Resource Identifier. The generic term for all types of names and addresses that refer to objects on the World Wide Web. A URL is one kind of URI.

URL

Uniform Resource Locator. A URL, a form of URI, is a compact string representation of the location for a resource that is available via the Internet. It is also the text-string format Web clients use to encode requests to Oracle Application Server.

V

virtual host

See multiport.

virtual machine

See JVM.

virtual path manager

The dispatcher forwards requests to the virtual path manager. The virtual path manager maps a request to a cartridge type and passes this information back to the dispatcher. The virtual path manager also passes back authentication requirements to the dispatcher. The dispatcher can then contact the authentication server for authorization and forward the request on to the correct cartridge type.

virtual pathname

In a virtual file system, a virtual pathname is a synonym that the virtual file system maps to a file stored in the file system maintained by the host machine's operating system.

VM

See JVM.

W

wallet

A wallet is an abstraction used to store and manager security credentials for an individual entity. It implements the storage and retrieval of credentials for use with various cryptographic services.

WAO

Web Application Objects. WAO is a framework designed to provide support for building transactional Web-based applications. WAO provides a set of objects for you to interact with the runtime environment of Web applications. The objects are implemented as CORBA objects with IDL interfaces, and they encapsulate lower-level operations so that you can focus on developing content.

Web Application Objects

See WAO.

Web Application Server (WAS)

Oracle Web Application Server is the name of the product used in Release 3.x of the Application Server. In Release 4.0, the product name was changed to Oracle Application Server.

Web Listener

Web Listener refers to the Oracle Web Listener that is shipped with Oracle Application Server. When you set up Oracle Application Server, a listener is set up for general usage requests. You may use the Oracle Web Listener for this, or you may choose to use a third-party listener instead.

See also HTTP listener, third-party listener.

Web Request Broker

See WRB.

WebServer

Oracle WebServer is the name of the product used in Release 1.x and 2.x of the Application Server. In Release 3.0, the name was changed to Web Application Server; in Release 4.0, the product name was changed to Oracle Application Server.

Web Site

A group of nodes sharing the same Web Request Broker (WRB).

WRB

Web Request Broker. A CORBA-based mechanism that provides resource management in handling requests for applications deployed as cartridges on the server. It provides a common set of services for managing applications. These services include load balancing, automatic failure recovery, security, and transaction services. Together with the ORB services, the Web Request Broker services make up the middle layer of the Oracle Application Server architecture.

wrb.app

The wrb.app file is the main configuration file for the Web Request Broker (WRB). There is one wrb.app file for each web site. The file is located at: %ORACLE_HOME%\ows\admin*website*\wrb\wrb.app for NT installations or $ORAWEB_ADMIN/*website*/wrb/wrb.app for Unix installations. See WRB.

WRB API

A C API used for writing server-side Web applications using the WRB.

X

XLF

Extended Logfile Format. XLF is an improved format for HTTP server logins since it is extensible, permitting a wider range of data to be captured. XLF allows you to configure the logger to generate different statistics of HTTP requests such as the IP address of clients, methods of the HTTP requests and response headers such as user agent and accept.

Index

A

adapter, 2-3
application deployment, 4-1
application logic, 1-3
application-level parameters, 3-7
applications
  C++ CORBA, 3-5
  cartridge-based, 3-1
  CGI, 3-6, 4-2
  configuring, 3-7
  CORBA-based, 3-3
  deploying, 3-6
applications servers, 2-6
architecture, 2-1
authentication schemes, 4-5

C

C++ CORBA cartridge, 3-5
cartridge
  C++ CORBA, 3-5
cartridge servers, 2-6
cartridge-based applications, 3-1
cartridge-level parameters, 3-8
cartridges
  COBOL, 4-3
  definition, 2-5
  introduction, 2-5
  JServlet, 3-2
  LiveHTML, 3-2
  ODBC, 3-3
  Perl, 3-3
  PL/SQL, 3-2
client, 1-1
clients, thin, 1-2
COBOL cartridge, 4-3
Common Gateway Interface (CGI)
    applications, 3-6, 4-2
components, 2-4
  ORB, 2-4
content service, 4-10
CORBA, 1-3
  applications, 3-3
  CORBA/IIOP requests, 3-9
CORBA applications, 4-2

D

database server, 1-1
databases
  ODBC, 4-4
  Oracle, 4-3
deployment, application, 4-1
Developer, 4-2
dispatcher, 2-3

E

ECO/Java
  applications, 3-4
EJB
  applications, 3-3, 3-10
  server, 3-10
Enterprise JavaBeans applications, 3-3, 3-10

F

failure detection, 4-7
failure recovery, 4-7

H

HTTP
  HTTP requests, 3-8
HTTP listener layer, 2-2
HTTP server, 2-2

I

ICX service, 4-9
Inter-ORB Protocol (IIOP), 1-3
introduction, 3-2
invoking JServlets, 3-2

J

Java
  Java stored procedures, 1-3
Java Servlet API Specification, 3-2
JDeveloper, 4-2
JServlet cartridge, 3-2

L

legacy applications, 4-3
listener, 2-2
listener layer, 2-2
LiveHTML cartridge, 3-2
load balancing, 4-8

M

middle tier, 1-3
multi-node installation, 4-7

O

Object Management Group(OMG), 1-3
ODBC cartridge, 3-3
ODBC databases, 4-4
OMG, 1-3
Oracle Application Server
  architecture, 2-1
  Manager, 4-6
  security, 4-5
  services, 4-8
  site management, 4-8
  system management, 4-6
Oracle Application Server layer, 2-3
Oracle Application Server layer components, 2-4
Oracle database, 4-3
Oracle Developer, 4-2
Oracle Wallet Manager, 4-6
ORB, 1-3
ORB components, 2-4
overview, 3-2

P

parameters
  application-level, 3-7
  cartridge-level, 3-8
Perl cartridge, 3-3
pl2java utility, 4-3
PL / SQL
  cartridge, 3-2
  stored procedures, 1-3
processes, 2-4

R

restriction schemes, 4-5
RM proxy, 3-10

S

secure sockets layer (SSL), 4-6
security, 4-5
  Oracle Wallet Manager, 4-6
  SSL, 4-6
session service, 4-9
site management, 4-8
SSL
  security schemes, 4-5

T

thin clients, 1-2
third-party tools, 4-3
three-tier computing model, 1-2
transaction service, 4-9
two-tier computing model, 1-1

V

virtual path manager, 2-3